NASA TECHNICAL NOTE

NASA TN D-4917

c.1

LOAN COPY: RET
AFWL (WLIL
KIRTLAND AFB, I

0131677

TECH LIBRARY KAFB, NM

NASA TN D-4917

COMPUTE — A TIME-SHARING DESK CALCULATOR PROGRAM

by Paul Swigert

Lewis Research Center
Cleveland, Ohio

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • NOVEMBER 1968

TECH LIBRARY KAFB, NM

0131677

COMPUTE - A TIME-SHARING DESK CALCULATOR PROGRAM

By Paul Swigert

Lewis Research Center
Cleveland, Ohio

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151 – CFSTI price $3.00

ABSTRACT

COMPUTE is a computer program, written primarily in FORTRAN IV, which operates under the IBM 360/67 Time-Sharing System (TSS). COMPUTE allows the TSS user to perform various numerical calculations without writing FORTRAN programs. The user may thus solve various numerical problems while at the TSS terminal by simply communicating with COMPUTE. This report is intended to introduce the user to COMPUTE. It is anticipated that the user will obtain from this introduction the information necessary to begin using COMPUTE. More specific information about the capabilities of COMPUTE can be derived from the actual use of the program. This is possible because of numerous error, warning, and information messages generated by COMPUTE and printed at the terminal.

CONTENTS

	Page
SUMMARY	1
INTRODUCTION	1
PRELIMINARY CONCEPTS	2
DEFINITION OF TERMS	4
COMPUTE STATEMENTS	5
Name Definition Statements	5
Values	5
User functions	6
Procedures	6
User programs	7
System functions	8
Output Statements	9
Value output	9
Nonvalue output	10
Erase Statements	10
Numerical Integration Statement	11
Return Statement	11
CONCLUDING REMARKS	11
APPENDIXES	
A - SUMMARY OF COMPUTE STATEMENTS AND ABBREVIATIONS	13
B - ANNOTATED COMPUTE LISTING	15

COMPUTE - A TIME-SHARING DESK CALCULATOR PROGRAM

by Paul Swigert

Lewis Research Center

SUMMARY

COMPUTE is a computer program written primarily in FORTRAN IV and operates under the IBM 360/67 Time Sharing System (TSS). COMPUTE allows the TSS user to perform various numerical calculations without writing FORTRAN programs. The user may thus solve various numerical problems while at the TSS terminal by simply communicating with COMPUTE.

This report is intended to introduce the user to COMPUTE. It is anticipated that the user will obtain from this introduction, information necessary to begin using COMPUTE. More specific information about the capabilities of COMPUTE can be derived from the actual use of the program. This is possible because of numerous error, warning, and information messages generated by COMPUTE and printed at the terminal.

INTRODUCTION

Computer time-sharing, or the simultaneous use of the computer by many users, allows the computer to be used in a variety of new ways. One of these new applications makes the computational power of a large computer available to the nonprogrammer, such as a research scientist, engineer, or mathematician. These personnel often need solutions to mathematical problems that are too lengthy for a desk calculator and yet simple enough so that the time taken to describe the problem to a programmer is greater than warranted.

Because of the unique man-machine dialogue provided for in time-sharing, the user may correct mistakes as they occur. This interaction eliminates the time involved in submitting several conventional batch computer jobs to correct errors. This time sometimes is measured in days, even for simple programs.

Another advantage to solving one's own problem with time-sharing is that the problem parameters may be varied dynamically by the user. That is, if an unexpected result is obtained in the solution of a problem, the causes may be investigated immediately and corrected or resolved. This approach to problem solving is completely absent in a non-time-sharing environment.

To provide this service to personnel who have had no, or limited, experience with computers and also to computer programmers, an interpretive on-line computer language was developed. Interpretive means that the language statements are executed or stored primarily in the form in which they are entered. On-line refers to the direct communication between the user and the computer.

This language, which consists of various statements, is the input to a computer program. This program, written primarily in FORTRAN, is called COMPUTE. The user need not distinguish between the physical computer, the program COMPUTE, or the language. He only needs to know a few statements and how to write FORTRAN-like expressions. Because of extensive error and information messages, the user is able to learn the language by using it.

Along with providing a nonprogrammer with computational facilities, such as function evaluation, algebraic expression evaluation, looping, and integration, COMPUTE is flexible enough to allow a programmer to expand COMPUTE's facilities. This is made possible by COMPUTE's ability to reference programs written by the user. These user programs may be anything from matrix manipulation programs to microfilm plotting programs.

The purpose of this report is to describe the use of COMPUTE. It is not intended to be a description of the COMPUTE program, but rather a user's manual. Some of the material in the report will be elementary for programmers, while some will seem advanced to the nonprogrammer. The potential user should not be too concerned with concepts he does not understand. The real advantage of a language of this type is that the user can learn it by experimenting. He should feel free to try his ideas and let COMPUTE point out errors and inconsistencies.

The next section gives some basic information about the overall use of COMPUTE. Terms used in describing COMPUTE commands are listed in the section DEFINITION OF TERMS. COMPUTE commands and input statements are then presented. Appendix A gives a summary of COMPUTE commands and statements with command abbreviations. The abbreviations may be used as if they were the command they represent. Appendix B is an annotated example of a COMPUTE terminal session.

PRELIMINARY CONCEPTS

COMPUTE may be accessed through a remote terminal connected to a central computer. The terminal may be one of many different types; for example, a typewriter, a teletype machine, and a CRT (cathode ray tube) with typewriter input. Each of these terminals will probably have its own method of entering, correcting, or canceling a line of input. Because of these differences, the physical operation of entering a line is not discussed. The reader should be aware, however, that the basic unit of input is one line of characters.

After the user has "logged on," that is, identified himself to the system, the first line to be entered is

RUN COMPUTE

The system will then print some information, and COMPUTE will print

ENTER USE KEYWORD

The use keyword is a parameter of COMPUTE that is used for special applications. The use keyword is of minor concern to a COMPUTE user and should be defaulted by entering a line of blanks (i.e., simply press the return key). COMPUTE will now print

INITIALIZATION COMPLETE
READY

The first line indicates that COMPUTE is in an initial state. The READY means that COMPUTE is now ready to accept user input. The READY is not printed when COMPUTE is waiting for a response, from the user, for some information needed to complete the current task.

Since the basic unit of input to COMPUTE is one line, provision is made for continuation lines. If the last character of any line is a vertical bar (|), the next line is considered to be a continuation. The maximum number of lines is three. Therefore, only two consecutive continuation lines are allowed.

Comments may be included anywhere in any line by enclosing the comments in apostrophes. For example, 'THIS IS A COMMENT.' would be considered as a comment by COMPUTE. Spaces or blanks are ignored by COMPUTE and may be placed anywhere in the input line.

COMPUTE will automatically inform the user of commands that are used incorrectly, of errors in expressions, and of various limits if they are exceeded. Therefore, the user does not need to remember many restrictions and forms. This is advantageous to the occasional user. The new user also benefits since he can let COMPUTE teach him its language by supplying information about itself.

DEFINITION OF TERMS

Name: One to eight alphameric characters, the first of which must be alphabetic. There are five types of names depending on how they are defined to COMPUTE. The method by which these names are defined to COMPUTE is discussed in the next section. The five types are

(1) Value names
(2) User function names
(3) Procedure names
(4) User program names
(5) System function names

Examples: L5
JOB1
ALPHA

Value: A number whose magnitude is between the approximate limits of 10^{75} and 10^{-75}, or zero. These numbers may be written in scientific notation by replacing the $\times 10^{\pm NN}$ by E±NN. The plus sign may be omitted. For example, 1.63×10^{-23} is written as 1.63E-23. All computation is performed with about six decimal digits. Number inputs to COMPUTE are rounded to this accuracy.

Examples: 21
.203
5E3
5.0E-3

Expression: Values, names, and function references (user or system) combined by arithmetic operation symbols and parentheses. The arithmetic operators are

(1) Exponentiation, **
(2) Multiplication, *
(3) Division, /
(4) Addition, +
(5) Subtraction, -

Where parentheses are omitted or where the entire arithmetic expression is enclosed within a single pair of parentheses, the order in which the operations are performed is as follows:

(1) Evaluation of functions
(2) Exponentiation, **
(3) Multiplication and division, * and /

(4) Addition and subtraction, + and -

In addition, if two operators of the same level are used consecutively, the operations are performed from left to right.

Parentheses may be used in arithmetic expressions, as in algebra, to specify the order in which the arithmetic operations are to be performed. Where parentheses are used, the expression within the parentheses is evaluated before the result is used. Example: A**(-1.5)*SIN(BETA/3.6)/(X+Y) is equivalent to

$$\frac{a^{-1.5} \sin\left(\frac{\beta}{3.6}\right)}{x + y}$$

COMPUTE STATEMENTS

This section gives descriptions and examples of all valid COMPUTE statements and commands. Most of the commands have abbreviations that may be used in place of the full command. These abbreviations along with a summary of COMPUTE statements are given in appendix A.

Name Definition Statements

COMPUTE allows the user storage of and/or access to certain types of information. This information is grouped into five categories:

(1) Values
(2) User functions
(3) Procedures
(4) User programs
(5) System functions

Each piece of information to be saved or accessed, except the system supplied functions, is given a name by the user. The user may use any name he chooses except the reserved system function names, such as SIN, COS, and ABS. Each of the categories are discussed separately in the following paragraphs:

Values. - Values are defined to the program by typing a name, an equal sign, and an expression. The program will evaluate the expression, save the result, and give the result the name appearing on the left of the equal sign.

General form: name = expression

Examples: A = 3.0
B = A*(26+2.8)
ALPHA = SIN(2.8)
Z23 = Z*(X+Y)

User functions. - User functions are defined to the program by typing a name (the name to be assigned to the function), argument names separated by commas and enclosed in parentheses, an equal sign, and an expression that contains the argument names. (Argument names are dummy variables and are meaningful only in the function definition.) The program will save the function and give it the name appearing to the left of the parentheses enclosing the arguments. This user function may then be evaluated, by use in an expression, as many times as desired.

General form: names(name1, name2, name3, . . .) = expression

Examples:
```
CUBRT(X) = X**0.333333
F(X) = A*X**2+B*X+C
DF(X) = 2*A*X+B
G(X) = X-F(X)/DF(X)
NR5(X) = G(G(G(G(G(X)))))
SIN2(Z) = SIN(Z)**2
SUMSQ(X, Y) = X**2+Y**2
```

(The function NR5, when evaluated, will give a value equal to five Newton-Raphson iterations on the function F, where the argument to NR5 is the initial guess.)

Procedures. - A procedure is a group of value definitions and/or value output statements that are to be used in a looping or iterative process. The general form of defining procedures to the program is

```
BEGIN (name)
Any number of value definitions and/or value output statements.
END or END (expression 1 > or < expression 2)
```

COMPUTE will save the statements between the BEGIN and END statements and give them the name that appears in the parentheses of the BEGIN statement. These statements will be numbered automatically by COMPUTE. The numbers are appended to the READY statements printed between the BEGIN and END statements. Two forms of the END statement are permitted. The first one, where no parentheses appear, assumes that the looping will terminate after a maximum number (specified by the user) of iterations has been reached. The second form allows the user to end the looping when the conditional expression inside the parentheses is satisfied or after a maximum number of iterations, whichever occurs first. No provision has been made to allow the alteration or insertion of statements in procedures.

```
Example: BEGIN (ALPHA)
         X = X+1
         Y = SQRT (X)
         X = ?
         Y = ?
         END (X>25)
```

To perform the operations stored in procedures the name of the procedure and the maximum number of iterations must be supplied. The general form of supplying this information is

DO (name*value)

or

DO (name)

Here, name refers to the name of a procedure and value to the maximum number of iterations desired. If the second form is used, value is assumed to be 1.

```
Examples: DO (INT*20)
          DO (BETA*5)
          DO (NEWTON)
```

User programs. - User programs are FORTRAN or assembly language subprograms that have been compiled or assembled before running COMPUTE. Access to a function subprogram is achieved by simply using the function name, with the proper number of arguments, in an expression. A call to subroutine is made by typing the subroutine name along with the arguments, if any, enclosed in parentheses. In this manner the user defines to COMPUTE the name as a user program name.

The subprograms are free to reference data, call other subprograms, and perform input and output like any TSS subprogram. The subprograms are restricted, however, from passing data back to COMPUTE through the calling vector. Data may be passed to COMPUTE only as the result of a function subprogram. Since COMPUTE performs all its calculations in floating point arithmetic, it makes no sense to consider subprograms that require arguments or yield function results in any other number type.

In the following examples, assume that SUB1 and SUB2 are subroutines and that F1 and F2 are functions that have been previously compiled or assembled. The first example would cause subroutine SUB1 to be loaded and called. The execution of a return statement in SUB1 would return control to COMPUTE. In the second example, function subprogram F1 would be loaded and called with two arguments, X and Y. Subroutine SUB2 would then be loaded and called with three arguments, X, Y, and the value obtained from funcation F1. The third example would load F2 and make three calls to it. The first call would have Z as the argument. The second call would use the value obtained from the first call, and the last call would use the value from the second call as the argument.

Examples: SUB1

SUB2(X, Y, F1(X, Y))

A = F2(F2(F2(Z)))

System functions. - Certain commonly used function have been defined to COMPUTE and are always available to the user. The names of these functions are system function names. The system function names are reserved names in that the user may not use them except to refer to the functions they represent.

A list of system functions available to the user, at this writing, appears in table I. Two facts about the system functions in table I are worth noting: (1) all trigonometric functions either accept radian measure as their argument or yield radian measure as their functional value, and (2) the integration function INT is simply a function of three argu-

TABLE I. - LIST OF AVAILABLE SYSTEM FUNCTIONS

Name	Definition	Argument range
EXP	Exponential	X < 174.673
LN	Natural logarithm	X > 0
LOG	Common logarithm	X > 0
SIN	Sine	|X| < (2**18)*PI
COS	Cosine	|X| < (2**18)*PI
TAN	Tangent	|X| < (2**18)*PI
ARCSIN	Arcsine	|X| < 1
ARCCOS	Arccosine	|X| < 1
ARCTAN	Arctangent	No restriction
SINH	Hyperbolic sine	X < 174.673
COSH	Hyperbolic cosine	X < 174.673
TANH	Hyperbolic tangent	No restriction
SQRT	Square root	X > = 0
ERF	Error function	No restriction
ERFC	Complemented error function	No restriction
GAMMA	Gamma function	2**(-252) < X < 57.574
LNGAMMA	Natural logarithm of gamma function	0 < X < 4.2913E + 73
ABS	Absolute value	No restriction
INT	Integration: 3 arguments (USR function, limit, limit)	No restriction

ments, the user function name, an expression for the lower limit, and an expression for the upper limit.

Any system function may be used in user function definitions, thus allowing the user to build quite complicated functions. The system functions may, of course, be nested just as user functions and user program functions.

Additions and alterations may be made to the system functions from time to time. The user is, therefore, advised to obtain a current list by occasionally using the DUMP command. The DUMP command is discussed in the section Nonvalue output.

Output Statements

There are two categories of output statements recognized by COMPUTE:

(1) Value output

(2) Nonvalue output

The value output category allows the user to print results of numerical operations, while the nonvalue category allows the user to print user functions, procedures, and system function names. Only statements in the value output category are allowed in procedures. The value output statements are discussed first.

Value output. - Value output is divided into three general forms. A description of these forms follows:

(1) Form 1: To obtain the value assigned to a value name, the user types the value name followed by an equal sign and a question mark.

General form: name = ?

Examples: A = ?

ROOT = ?

(2) Form 2: The user may obtain the result of an expression by typing the expression followed by an equal sign and a question mark.

General form: expression = ?

Examples: 6.0**0.5 = ?

SIN(2.6) = ?

A+B*COS(C) = ?

(3) Form 3: For output of several values assigned to value names, the PRINT command is available. The names that appear in this command must be value names.

General form: PRINT (name1, name2 . . .)

Examples: PRINT (A, B, C)

PRINT (ROOT, ANSWER)

Nonvalue output. - Nonvalue output is also divided into two general forms. A description of these forms follows:

Form 1: To obtain information about a nonvalue name (i. e., a user function, procedure, program, or system function name), the user types the name followed by an equal sign and a question mark. COMPUTE will then respond with a description of the name type.

General form: name = ?

Examples: ALPHA = ?

SIN = ?

SQRT = ?

Form 2: To obtain a listing of all names of a certain type, the user types the command DUMP followed by an option enclosed in parentheses. If the user specifies value names to be listed, the corresponding values are also listed.

General form: DUMP (option)

$$\text{Where option} = \begin{cases} \text{VALUES} \\ \text{USR FUNCTIONS} \\ \text{SYS FUNCTIONS} \\ \text{PROCEDURES} \end{cases}$$

Examples: DUMP (VALUES)

DUMP (USR FUNCTIONS)

DUMP (PROCEDURES)

DUMP (SYS FUNCTIONS)

Erase Statements

Two commands are provided that allow the user to delete, from the program, names that have been defined by the user.

The first command deletes all names, except system function names and user program names, from COMPUTE. This command also allows the user to enter a new use keyword if desired.

General form: RESTART

Example: RESTART

The second command deletes specific names from the program. To accomplish this the user types the command ERASE followed by the names to be deleted enclosed by parentheses. These names may include any name that has been defined by the user, except user program names. This command does not allow the user to erase specific lines of a procedure, but will erase complete procedures.

General form: ERASE (name1, name2, . . .)

Examples: ERASE (F, DF)
ERASE (A, B, ALPHA)

Numerical Integration Statement

Numerical integration of user functions is provided for in COMPUTE through the INTEGRATE command. After the INTEGRATE command is entered, COMPUTE prompts the user for the user function name or user function definition and for the limits of integration. The answer is then printed. A warning is printed if the answer is not accurate to five significant figures. The INTEGRATE command is simply a different form of the INT function described in the section Name Definition Statements. Results obtained from the two forms will be identical.

General form: INTEGRATE

Example: INTEGRATE

Return Statement

A command is provided by COMPUTE that allows the user to return to the TSS system.

General form: STOP

Example: STOP

CONCLUDING REMARKS

The computer time-sharing program described in this report promises to make some of the computational power of large computers available to noncomputer personnel. Complex desk-type calculations are made simple and fast. The program has been made flexible enough so that additions are easy to make. When new facilities are added to the computer system, COMPUTE may be modified easily to incorporate them.

Extensions of COMPUTE that are presently being considered are (1) plots of user functions on a CRT (cathode ray tube) or on microfilm, (2) algebraic manipulation, and

(3) a differential equations solver. With these added features COMPUTE will become an even more powerful tool for on-line problem solving.

Lewis Research Center,
National Aeronautics and Space Administration,
Cleveland, Ohio, October 3, 1968,
125-23-02-15-22.

APPENDIX A

SUMMARY OF COMPUTE STATEMENTS AND ABBREVIATIONS

This appendix contains a list of all valid COMPUTE statements, along with a brief description and abbreviation (if any). The abbreviation may be used in place of the full command. For a complete description of the commands, the text of this report should be consulted.

General form of statement	Description	Abbreviation
name = expression	Defines name as the value obtained from (expression).	None
name(name1, name2, . . .) = expression	Defines name as a user function with the function being (expression).	None
BEGIN (name)	Denotes the start of what is to be procedure (name).	B(name)
END or END (expression1 > or < expression2)	Denotes the end of the procedure started by a previous begin statement.	None
DO(name*value) or DO(name)	Causes statements stored in procedure (name) to be looped through a maximum number (value) or times.	None
name = ?	Causes the value stored in (name) or information about (name) to be printed.	None
expression = ?	Causes the value obtained from evaluation of (expression) to be printed.	None
PRINT (name1, name2, . . .)	Causes the values stored in (name1), (name2), . . . , etc. to be printed.	P(name1, name2, . . .)

General form of statement	Description	Abbreviation
DUMP (option)	Causes all names of the type specified in option to be printed.	D(option)
	option: VALUES	V
	USR FUNCTIONS	UF
	SYS FUNCTIONS	SF
	PROCEDURES	P
RESTART	Initializes COMPUTE by erasing all names except system and user program names. Allows the user to enter a new use keyword.	R
ERASE (name1, name2, . . .)	Erases the values, user functions, and procedures with the names (name1), (name2), . . . , etc.	E(name1, name2, . . .)
INTEGRATE	Numerically integrates user functions.	I
STOP	Returns user to TSS.	S

APPENDIX B

ANNOTATED COMPUTE LISTING

The listing presented in this appendix is from an actual COMPUTE session. The line numbers were added to aid the reader in following the description.

The user usually types in lower case, while the system and user programs type in upper case. This convention is terminal dependent, however.

Description

In the description the numbers refer to line numbers on the listing.

Line(s)	Description
1-2	User identifies himself to the time-sharing system.
3	User requests that program COMPUTE be run.
4	System message identifying input/output version.
5	COMPUTE requests use keyword from user.
6	User enters a line of blanks, by pressing return, to default this parameter.
7	COMPUTE acknowledges that the use keyword is valid and that COMPUTE is in an initial state.
8	COMPUTE indicates that it is ready to accept user input.
9-22	User defines a procedure named NEWTON. This procedure uses the Newton-Raphson technique to find the root of the user function F(X). The procedure will print the independent variable, functional values of the function F, the first derivative of F, and the correction factor H. The procedure will stop looping when the magnitude of the correction factor becomes less than the value of LIMIT. Note the numbering of statements contained in procedures. This procedure is six statements long.
23	The user requests that procedure NEWTON be looped through a maximum of 10 times.
24-25	COMPUTE asks the user to define an unknown variable and the user responds.
26-28	COMPUTE detects an error and informs the user.
29-33	User defines the two required functions and executes procedure NEWTON again.

Line(s)	Description
34-39	COMPUTE requests and user defines unknown values.
40	First line of output from NEWTON is printed by COMPUTE.
41-45	The last unknown value is defined, and COMPUTE continues to loop through NEWTON.
46-47	COMPUTE indicates that looping was stopped because the condition in the END statement was met.
48-56	User redefines the independent variable and initiates NEWTON again. An error is detected and reported by COMPUTE and verified by the user.
57-59	User changes the independent variable and initiates NEWTON.
60-69	Output generated by NEWTON.
70-71	Looping was terminated because of maximum iterations.
72-78	Since convergence has not been achieved, the user continues the looping by issueing another DO command. COMPUTE continues to loop in NEWTON until the condition in the END statement is met.
79-119	Illustration of the different options in the DUMP command.
120	User issues the INTEGRATE command.
121-126	COMPUTE prompts user for needed information and prints the answer.
127-129	User checks COMPUTE answer for the integral.
130-141	User integrates and checks the result for another function. In this example the user defines the function instead of supplying a function name.
142-145	The user makes use of the INT function to define Dawson's integral: $$D(X) = e^{-x^2} \int_0^x e^{t^2}\, dt$$ Note that the definition requires two user functions since the INT function requires a user function name be its first argument.
146-155	A second procedure is defined. This procedure, DAWSON will use the Dawson's integral function just defined and print a table.
156-160	The user initializes X and DELTAX and initiates the procedure DAWSON.
161-171	The table of Dawson's integral generated by the procedure is printed.

Line(s)	Description
172-173	COMPUTE informs the user that execution of the procedure was terminated because of the iteration count.
174-182	The user makes two references to a user supplied function subprogram. The subprogram MAX returns as its functional value the maximum of the supplied arguments. Note that the second reference does not cause the program loaded message to be printed. This is because the program is only loaded once.
183-188	User requests and obtains a printout of all value names he has defined.
189-192	User issues a STOP command for COMPUTE and a LOGOFF command for the time-sharing system. The system acknowledges the STOP and LOGOFF.

Sample COMPUTE Listing

```
#B001 LOGON  09/10/68 15:19
xxpaul,,n,m,ps
run compute
FIO VERSION JAN 1,1968__
ENTER USE KEYWORD.

INITIALIZATION COMPLETE.
READY
begin(newton)
READY   1
fx = f(x)
READY   2
dfx = df(x)
READY   3
h = -fx/dfx
READY   4
print(x,fx,dfx,h)
READY   5
x = x+h
READY   6
end(abs(h)<limit)
READY
do(newton•10)
       X UNKNOWN.  ENTER NUMBER OR PRESS RETURN TO CANCEL.
3
FUNCTION         F UNKNOWN.
ERROR IS IN STATEMENT   1 OF PROCEDURE   NEWTON
READY
f(x) = a+b*x+c*x**2
READY
df(x) = b+2*c*x
READY
do(newton*10)
       A UNKNOWN.  ENTER NUMBER OR PRESS RETURN TO CANCEL.
-20
       B UNKNOWN.  ENTER NUMBER OR PRESS RETURN TO CANCEL.
8.0
       C UNKNOWN.  ENTER NUMBER OR PRESS RETURN TO CANCEL.
1
        X=   3.00000            FX=   13.0000         DFX=   14.0000            H= -0.928571
    LIMIT UNKNOWN.  ENTER NUMBER OR PRESS RETURN TO CANCEL.
.00000001
        X=   2.07143            FX=  0.862241         DFX=   12.1429            H= -0.710080E-01
        X=   2.00042            FX=  0.503540E-02     DFX=   12.0008            H= -0.419587E-03
        X=   2.00000            FX=  0.0              DFX=   12.0000            H=  0.0
END OF DO.  0.0         < 0.100000E-07
READY
x=-4
READY
do(newton*10)
DIVIDE BY ZERO.
ERROR IS IN STATEMENT   3 OF PROCEDURE   NEWTON
READY
df(x)=?
=  0.0
READY
x=-4.01
```

```
READY
do(newton*10)
        X=  -4.01000            FX=  -35.9999           DFX= -0.199986E-01        H=  -1800.13
        X=  -1804.13            FX=  0.324045E 07       DFX=  -3600.27            H=   900.057
        X=  -904.078            FX=   810104.           DFX=  -1800.16            H=   450.019
        X=  -454.059            FX=   202517.           DFX=  -900.118            H=   224.989
        X=  -229.070            FX=   50620.3           DFX=  -450.139            H=   112.455
        X=  -116.615            FX=   12646.1           DFX=  -225.230            H=   56.1475
        X=  -60.4673            FX=   3152.55           DFX=  -112.935            H=   27.9149
        X=  -32.5524            FX=   779.239           DFX=  -57.1048            H=   13.6458
        X=  -18.9066            FX=   186.207           DFX=  -29.8132            H=   6.24578
        X=  -12.6608            FX=   39.0100           DFX=  -17.3217            H=   2.25210
END OF DO.  10 ITERATIONS.
READY
do(newton*10)
        X=  -10.4087            FX=   5.07190           DFX=  -12.8175            H=  0.395702
        X=  -10.0130            FX=  0.156586           DFX=  -12.0261            H=  0.130205E-01
        X=  -10.0000            FX=  0.167847E-03       DFX=  -12.0000            H=  0.139872E-04
        X=  -10.0000            FX=  0.0                DFX=  -12.0000            H=  0.0
END OF DO.  0.0          < 0.100000E-07
READY
dump(values)
DUMP OF VALUES.
        X=  -10.0000             A=  -20.0000            B=   8.00000             C=   1.00000
       FX=  0.0                DFX=  -12.0000            H=  0.0              LIMIT=  0.100000E-07
READY
dump(usr functions)
DUMP OF USER FUNCTION NAMES.
      F        1 ARGUMENTS.
     DF        1 ARGUMENTS.
READY
dump(procedures)
DUMP OF PROCEDURE NAMES.
  NEWTON
READY
dump(sys functions)
LIST OF AVAIABLE FUNCTIONS.
    NAME       DEFINITION               ARGUMENT RANGE
    EXP        EXPONENTIAL              X<174.673
    LN         NATURAL LOGARITHM        X>0
    LOG        COMMON LOGARITHM         X>0
    SIN        SINE                     |X|<(2**18)*PI
    COS        COSINE                   |X|<(2**18)*PI
    TAN        TANGENT                  |X|<(2**18)*PI
    ARCSIN     ARCSINE                  |X|<1
    ARCCOS     ARCCOSINE                |X|<1
    ARCTAN     ARCTANGENT               NO RESTRICTION
    SINH       HYPERBOLIC SINE          X<174.673
    COSH       HYPERBOLIC COSINE        X<174.673
    TANH       HYPERBOLIC TANGENT       NO RESTRICTION
    SQRT       SQUARE ROOT              X>=0
    ERF        ERROR FUNCTION           NO RESTRICTION
    ERFC       COMPLEMENTED             NO RESTRICTION
               ERROR FUNCTION
    GAMMA      GAMMA FUNCTION           2**(-252)<X<57.574
    LNGAMMA    NATURAL LOGARITHM        0<X<4.2913E+73
               OF GAMMA FUNCTION
    ABS        ABSOLUTE VALUE           NO RESTRICTION
    INT        INTEGRATION              NO RESTRICTION
               3 ARGUMENTS
               (USR FUNCTION,LIMIT,LIMIT)
READY
integrate
ENTER USER FUNCTION NAME, DEFINE USER FUNCTION, OR PRESS RETURN TO CANCEL.
df
ENTER LOWER LIMIT, A COMMA, UPPER LIMIT, OR PRESS RETURN TO CANCEL.
0,5.0
 THE INTEGRAL IS       65.0000
```

```
READY
f(5)-f(0)=?
=   65.0000
READY
integrate
ENTER USER FUNCTION NAME, DEFINE USER FUNCTION, OR PRESS RETURN TO CANCEL.
g(x)=sin(x)
ENTER LOWER LIMIT, A COMMA, UPPER LIMIT, OR PRESS RETURN TO CANCEL.
0.0,3.141593
 THE INTEGRAL IS       2.00000
READY
cos(0)-cos(pi)=?
      PI UNKNOWN.  ENTER NUMBER OR PRESS RETURN TO CANCEL.
3.141593
=   2.00000
READY
d(x)=exp(-x**2)*int(e2,0,x)
READY
e2(t) = exp(t**2)
READY
begin(dawson)
READY   1
dx = d(x)
READY   2
print(x,dx)
READY   3
x = x+deltax
READY   4
end
READY
x = 0
READY
deltax=.2
READY
do(dawson*11)
        X=  0.0              DX=  0.0
        X=  0.200000         DX=  0.194751
        X=  0.400000         DX=  0.359943
        X=  0.600000         DX=  0.474763
        X=  0.800000         DX=  0.532101
        X=  1.000000         DX=  0.538079
        X=   1.20000         DX=  0.507273
        X=   1.40000         DX=  0.456507
        X=   1.60000         DX=  0.399939
        X=   1.80000         DX=  0.346771
        X=   2.00000         DX=  0.301338
END OF DO.  11 ITERATIONS.
READY
max(a,b,c)=?
PROGRAM      MAX LOADED.
=   8.00000
READY
maximum = max(a**2,b**2,c**2)
READY
maximum=?
 MAXIMUM=   400.000
READY
dump(values)
DUMP OF VALUES.
        X=   2.20000         A=  -20.0000          B=   8.00000          C=   1.00000
       FX=  0.0            DFX=  -12.0000          H=  0.0          LIMIT=  0.100000E-07
       PI=   3.14159    DELTAX=  0.200000         DX=  0.301338   MAXIMUM=   400.000
READY
stop
CHCIW EXIT   IN USER PROGRAM
logoff
B007 LOGOFF ACCEPTED 09/10/68 AT 15:56.
```


NASA-Langley, 1968 —— 8 E-4681

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
WASHINGTON, D. C. 20546

OFFICIAL BUSINESS

FIRST CLASS MAIL

POSTAGE AND FEES PAID
NATIONAL AERONAUTICS AND
SPACE ADMINISTRATION

050 001 33 51 3DS 68304 00903
AIR FORCE WEAPONS LABORATORY/AFWL/
KIRTLAND AIR FORCE BASE, NEW MEXICO 87117

ATT E. LOU BOWMAN, ACTING CHIEF TECH. LI

POSTMASTER: If Undeliverable (Section 158 Postal Manual) Do Not Return

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include conference proceedings, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Notes, and Technology Surveys.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION DIVISION
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
Washington, D.C. 20546